

데이터베이스시스템 과제 2-1

<dbms로 만들기>

2017030519 홍유진


<step 1>

- 관리자 모드에 새로운 학생 '홍길동' 추가하기

1. 한양관리자로 로그인하기 위해 user에 admin을 추가한다.

```
1 create user admin password '1111' login
```

2. student table을 만든 후 students.csv에 있는 내용을 넣어준다.

 admin on postgres@hanyangpotal

```
1 create table student (  
2     sid varchar primary key,  
3     password varchar,  
4     sname varchar,  
5     sex varchar,  
6     major_id integer,  
7     tutor_id varchar,  
8     grade integer  
9 )
```

Data Output

Explain

Messages

Notifications

Query History

	sid character varying	password character varying	sname character varying	sex character varying	major_id integer	tutor_id character varying	grade integer
1	2009003125	125125125	정남아	female	44	2001032011	4
2	2010004052	39nnf2	김관유	male	4	2001032064	3
3	2010499349	2bn4	이현주	female	5	2001032031	4
4	2011004533	35234	한준희	male	3	2001032070	3
5	2011040404	x17171771	김다미	female	15	2001032068	3
6	2012004003	banila	김동관	male	33	2001032030	3
7	2012004203	qwe123	오든솔	male	5	2001032007	3
8	2012030303	arandomkey	최다비드				
9	2012394929	3425nn5	이지은				

✓ Successfully run. Total query runtime: 443 msec. 20 rows

이런식으로 들어가 있는 것을 확인할 수 있다.

3. 홍길동의 정보 추가하기

```
admin on postgres@hanyangpotal
1 insert into student values('2016001234','xxx','홍길동','male',6,'1999002345',1)
```

	sid character varying 2013004394	password character varying goooooy	sname character varying 관리자	sex character varying male	major_id integer 1	tutor_id character varying 2001032004	grade integer 2
13	2013030128	food	윤지형	male	44	2001032078	2
14	2013040051	zeroone234	최아람	female	19	2001032078	2
15	2014001303	192939	김다현	female	3	2001032009	1
16	2014002331	wer234	이상덕	male	2	2001032008	1
17	2014005004	hexahed	정두호	male	5	2001032081	1
18	2014019493	zerozero	임지훈	male	17	2001032003	1
19	2014040404	donkey	권지현	female	22	2001032053	1
20	2014505050	helloworld	권희조	female	44	2001032078	1
21	2016001234	xxx	홍길동	male	6	1999002345	1

이렇듯 21번째 행에 홍길동이 추가된 것을 알 수 있다.

이처럼 새로운 학생을 추가하려면 insert를 이용해 추가할 수 있다.

- 관리자 모드에서 '한양인 주소록' 파일에 연락처 추가기능을 구현하고, '홍길동'의 연락처를 추가한다.

1. 한양인 주소록 테이블을 만든다.

```
admin on postgres@hanyangpotal
1 create table 한양인주소록 (
2     sid varchar primary key,
3     phone varchar,
4     email varchar
5 )
```

Data Output	Explain	Messages	Notifications	Query History
sid character varying	phone character varying	email character varying		
1	2013004394	01056256058	amaclead@gmail.com	
2	2010499349	01058712131	art@venere.org	
3	2014001303	01068220334	calbares@gmail.com	
4	2011040404	01078554051	donette.foller@cox....	
5	2014505050	01062384210	fletcher.flosi@yahoo...	
6	2014019493	01056662093	gladys.rim@rim.org	
7	2013040051	01064989926	gruta@cox.net	
8	2009003125	01095434668	jbutt@gmail.com	
9	2010004052	01022215104	josephine_darakjy@...	

그리고 값을 추가한다. (contacts.csv)

2. 홍길동을 한양인 주소록에 추가한다.

	admin on postgres@hanyangpotal
1	insert into 한양인주소록 values ('2016001234', '01088884444', 'hong@hanyang.ac.kr');
2	

	sid character varying	phone character varying	email character varying
11	2015001445	01089941847	kris@gmail.com
12	2012030303	01060294130	leota@hotmail.com
13	2011004533	01080969723	lpaprocki@hotmail.c...
14	2014005004	01096385105	meaghan@hotmail.c...
15	2013004004	01092039494	minna_amigon@yah...
16	2012004203	01094753033	mitsue_tollner@yah...
17	2012394929	01020374564	sage_wieser@cox.net
18	2012004003	01035039847	simona@morasca.co...
19	2014040404	01085931087	yuki_whobrey@aol.c...
20	2016001234	01088884444	hong@hanyang.ac.kr

20번행에 홍길동의 주소록이 추가된 것을 확인할 수 있다.

- 관리자 모드에서 연락처 변경 기능을 구현하고, 학생 '권희조'의 이메일을 'kwon@hanyang.ac.kr'로 변경한다.

1. sql문 활용

```
update 한양인주소록 h
set email =(select 'kwon@hanyang.ac.kr')
from student s
where s.sid = h.sid and s.sname = '권희조';
```

: update구문을 이용해 바꾼다.

- 관리자 모드에서 연락처 삭제 기능을 구현하고, 학생 '김다현'의 이메일을 번호 이메일만 삭제한다.

1. sql문 이용

	Data Output	Explain	Messages	Notifications	Query History
	sid character varying	phone character varying	email character varying		
1	2013004394	01056256058	amaclead@gmail.com		
2	2010499349	01058712131	art@venere.org		
3	2014001303	01068220334	calbares@gmail.com		
4	2011040404	01078554051	donette.foller@cox....		

김다현의 학생정보는 3번째이다. 즉, 학번은 2014001303이다.

× 찾기: 2014001303 이전 다음 옵션 ▼ 일치하는 항목이 없음

pgAdmin 4 File Object Tools Help

Browser Servers (2) PostgreSQL hanyang Da

admin on postgres@hanyangpotal

```

1 delete from 한양인주소록
2 where 한양인주소록.sid in (
3     select sid
4     from student s
5     where 한양인주소록.sid=s.sid and s.sname='김다현'
6 );
7 select * from 한양인주소록

```

	Data Output	Explain	Messages	Notifications	Query History
	sid character varying	phone character varying	email character varying		
1	2013004394	01056256058	amaclead@gmail.com		
2	2010499349	01058712131	art@venere.org		
3	2011040404	01078554051	donette.foller@cox....		
4	2014019493	01056662093	gladys.rim@rim.org		
5	2014505050	01062384210	kwon@hanyang.ac.kr		
6	2009003125	01095434668	jbutt@gmail.com		
7	2010004052	01022215104	josephine_darakjy@...		
8	2013030128	01058290113	kiley.caldarera@aol....		
9	2013001445	01089941847	kris@gmail.com		
10	2012030303	01060294130	leota@hotmail.com		
11	2011004533	01080969723	lpaprocki@hotmail.c...		
12	2014002331	01043813720	mattie@aol.com		
13	2014005004	01096385105	meaghan@hotmail.c...		

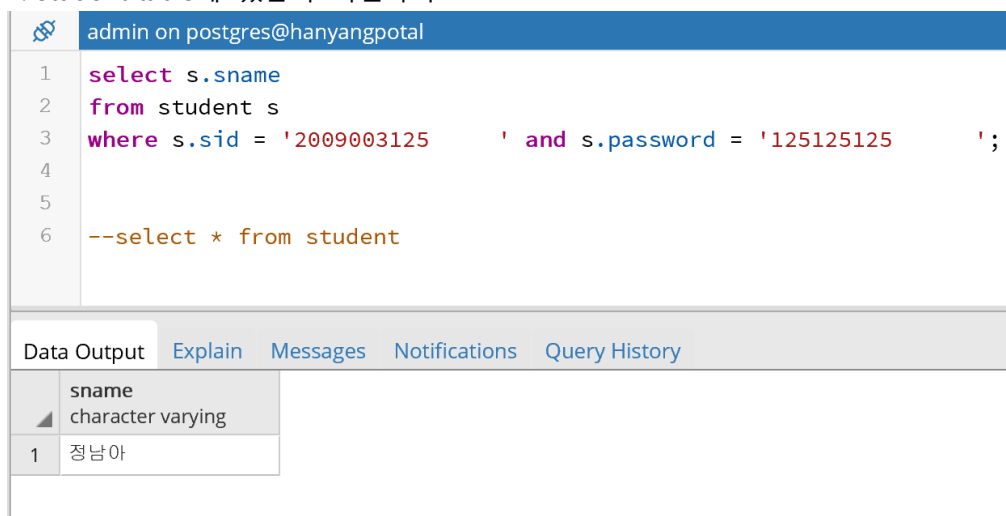
하지만 delete를 이용한 sql문을 실행했을 때 김다현의 연락처가 사라진 것을 확인할 수 있다.

<step 2>

- 이용자로 로그인하기

Admin database의 student table에서 이용자로 로그인하려는 사람이 있는지 확인한 후 있으면 로그인 부분에 추가한다.(이때 로그인의 id는 sid(학번)이다.) 만약에 등록 후 로그인을 할 때 비밀번호가 틀리면 프로그램에서 에러메세지를 출력할 것이다.

1. student table에 있는지 확인하기



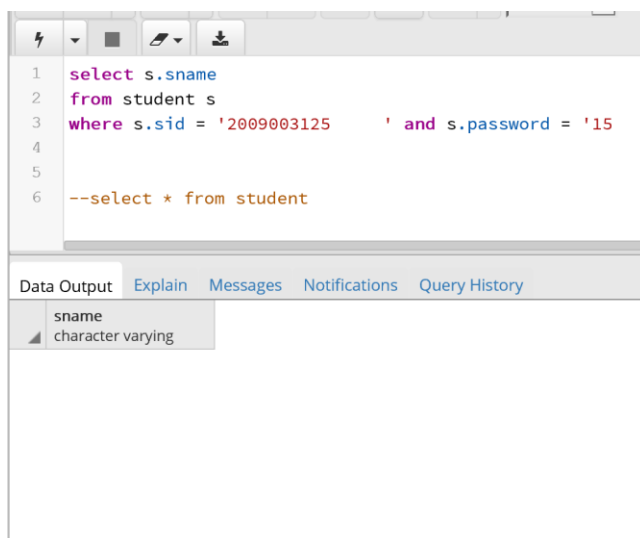
The screenshot shows a PostgreSQL query editor interface. The top bar indicates the user is 'admin on postgres@hanyangpotal'. The query editor contains the following SQL code:

```
1 select s.sname
2 from student s
3 where s.sid = '2009003125' and s.password = '125125125';
4
5
6 --select * from student
```

Below the query editor, there are tabs for 'Data Output', 'Explain', 'Messages', 'Notifications', and 'Query History'. The 'Data Output' tab is selected, showing a table with one row:

sname
정남아

:위의 sql문에서 정남아가 있으면 sname을 출력한다.



The screenshot shows a PostgreSQL query editor interface. The top bar indicates the user is 'admin on postgres@hanyangpotal'. The query editor contains the following SQL code:

```
1 select s.sname
2 from student s
3 where s.sid = '2009003125' and s.password = '15';
4
5
6 --select * from student
```

Below the query editor, there are tabs for 'Data Output', 'Explain', 'Messages', 'Notifications', and 'Query History'. The 'Data Output' tab is selected, showing a table with one row:

sname
정남아

만약 비밀번호가 틀리면 아무것도 출력하지 않을 것이다.

2. table에 존재하는 학생이고 비밀번호가 맞다면 login부분에 학생을 추가한다.

```
1 create user "2009003125" | password '125125125' login
```

```
1 SELECT * FROM PG_SHADOW;
```

Data Output Explain Messages Notifications Query History

	username name	usesysid oid	usecreatedb boolean	usesuper boolean	userepl boolean	usebypassrls boolean	passwd text	valuntil abstime	useconfig text[]
1	postgres	10	true	true	true	true	md5b15...	[null]	[null]
2	admin	24587	true	false	false	false	md5a27e...	[null]	[null]
3	200900312...	32875	false	false	false	false	md58cd4...	[null]	[null]
4	201300439...	32876	false	false	false	false	md52bb...	[null]	[null]
5	201400500...	32877	false	false	false	false	md5cca5...	[null]	[null]

우선 과제에서 수행하는 학생들을 로그인에 추가한다.

혹은 로그인에 추가하지않더라도 프로그램내에서 로그인이 실패하면 후에 내용을 실행하지 못하도록 구현하면 된다.

- business_card.zip의 연락처를 개인 주소록에 넣는다.

각각 개인주소록 table을 만들어준다.

```
create table grass_corp(  
    name varchar,  
    phone varchar primary key,  
    email varchar,  
    rank varchar  
)
```

(예시로 grass_corp만 보임)

	Data Output	Explain	Messages	Notifications	Query History
	name character varying	phone character varying	email character varying	rank character varying	
1	이상해씨	01023140011	bulbasaur@grass.p...	사장	
2	치코리타	01051522001	chikorita@grass.poke	부장	
3	주리비안	01064950101	snivy@grass.poke	과장	
4	나무지기	01032540033	sceptile@grass.poke	사원	

각각의 csv파일을 import해서 테이블안에 값을 넣어줌

(admin=# ``에서 admin은 데이터베이스의 이름을 임의로 admin으로 설정하였기 때문이다.)

후에 학생은 자신이 추가한 파일의 테이블(자신의 개인주소록)만 볼 수 있음

- 개인주소록에business_card_updated.zip 을 각각 반영

<ver1 : 원래파일에 업데이트된 부분을 추가할 때, 즉 원래파일에 없던 주소록이 업데이트 파일에 생겼을 때, 원래파일에 있던주소록에 업데이트파일엔 없어졌을 때>
(업데이트부분이 잘 이해가 안되 두가지 버전으로 만들어보았습니다.)

1. 업데이트할 파일을 불러오기 위해 temp 라는 테이블을 만든다.

```
create table temp(
  name varchar,
  phone varchar primary key,
  email varchar,
  rank varchar
)
```

2. grass 를 예시로 들어본다.

-1) temp 에 업데이트된 grass_corp.csv 의 정보를 넣는다.

	Data Output	Explain	Messages	Notifications	Query History
	name character varying	phone character varying	email character varying	rank character varying	
1	이상해풀	01023140011	ivysaur@grass.poke	이사	
2	치코리타	01051522001	meganium@grass.p...	사장	
3	리피아	01061344185	leafeon@grass.poke	부장	
4	주리비안	01064950101	snivy@grass.poke	과장	
5	나무지기	01032540033	sceptile@grass.poke	사원	

2. 원래파일에 업데이트된 번호가 없을 때

	Data Output	Explain	Messages	Notifications	Query History
	tname character varying	tphone character varying	temail character varying	trank character varying	
1	리피아	01061344185	leafeon@grass.poke	부장	

insert 문은 잠시 주석처리해주었다. select 문을 활용하여 전화번호가 있지 않은 곳을 찾을 수 있다.

	Data Output	Explain	Messages	Notifications	Query History
	name character varying	phone character varying	email character varying	rank character varying	
1	이상해씨	01023140011	bulbasaur@grass.p...	사장	
2	치코리타	01051522001	chikorita@grass.poke	부장	
3	주리비안	01064950101	snivy@grass.poke	과장	
4	나무지기	01032540033	sceptile@grass.poke	사원	
5	리피아	01061344185	leafeon@grass.poke	부장	

인서트 해주면 리피아가 추가된 것을 알 수 있다.

3. 번호는 같지만 다른 정보가 있을 때

admin on postgres@hanyangpotal				
1	update grass_corp			
2	set name = temp.name, email=temp.email, rank=temp.rank			
3	from temp			
4	where grass_corp.phone = temp.phone;			
5				
6				
7	--insert into grass_corp (name,phone,email,rank)			
8	--select t.name as tname,t.phone as tphone,t.email as temail,t.rank as trunk			
9	--from temp t			
10	--where t.phone <> all (select g.phone			
11	-- from grass_corp g			

Data Output	Explain	Messages	Notifications	Query History
name	phone	email	rank	
character varying	character varying	character varying	character varying	
1 이상해풀	01023140011	ivysaur@grass.poke	이사	
2 치코리타	01051522001	meganium@grass.p...	사장	
3 주리비안	01064950101	snivy@grass.poke	과장	
4 나무지기	01032540033	sceptile@grass.poke	사원	
5 리피아	01061344185	leafeon@grass.poke	부장	

update 문을 이용해 업데이트 해줬다. 그 결과 테이블이 제대로 나온 것을 확인할 수 있다.

4. 전체 sql 문

admin on postgres@hanyangpotal	
1	insert into grass_corp (name,phone,email,rank)
2	select t.name as tname,t.phone as tphone,t.email as temail,t.rank as trunk
3	from temp t
4	where t.phone <> all (select g.phone
5	from grass_corp g
6);
7	
8	update grass_corp
9	set name = temp.name, email=temp.email, rank=temp.rank
10	from temp
11	where grass_corp.phone = temp.phone;
12	
13	
14	
15	select * from grass_corp

먼저 2 번과정을 수행한후 3 번과정을 수행한다면 업데이트한 테이블을 얻을 수 있다. 위에 sql 문에서 table 명만 바꾸면 다른 테이블 또한 명령을 잘 수행할 수 있다.

<ver2> 업데이트파일로 원래파일을 아이에 바꿀 때

1. 업데이트할 파일을 불러오기 위해 temp 라는 테이블을 만든다.

```
create table temp(
  name varchar,
  phone varchar primary key,
  email varchar,
  rank varchar
)
```

2. grass 를 예시로 들어본다.

-1) temp 에 업데이트된 grass_corp.csv 의 정보를 넣는다.

Data Output	Explain	Messages	Notifications	Query History
	name character varying	phone character varying	email character varying	rank character varying
1	이상해풀	01023140011	ivysaur@grass.poke	이사
2	치코리타	01051522001	meganium@grass.p...	사장
3	리피아	01061344185	leafeon@grass.poke	부장
4	주리비안	01064950101	snivy@grass.poke	과장
5	나무지기	01032540033	sceptile@grass.poke	사원

3. 원래의 grass_corp table을 drop하고 temp의 있는 값을 복사해 새로 만든후 temp를 drop한다.

admin on postgres@hanyangpotal
1 drop table grass_corp cascade;
2 create table grass_corp (like temp including all);
3 insert into grass_corp(select * from temp);
4 drop table temp cascade;
5 select * from grass_corp
6

Data Output	Explain	Messages	Notifications	Query History
	name character varying	phone character varying	email character varying	rank character varying
1	이상해풀	01023140011	ivysaur@grass.poke	이사
2	치코리타	01051522001	meganium@grass.p...	사장
3	리피아	01061344185	leafeon@grass.poke	부장
4	주리비안	01064950101	snivy@grass.poke	과장
5	나무지기	01032540033	sceptile@grass.poke	사원

원래의 grass_corp table 의 내용이 업데이트 파일의 내용으로 변경된 것을 알 수 있다.

- 정남아, '윤인욱', '장두호'의 개인주소록에서 임의의 연락처 1 건을 삭제

1. 원래의 grass_corp

10 `select * from grass_corp`

	name character varying	phone character varying	email character varying	rank character varying
1	이상해풀	01023140011	ivysaur@grass.poke	이사
2	치코리타	01051522001	meganium@grass.p...	사장
3	리피아	01061344185	leafeon@grass.poke	부장
4	주리비안	01064950101	snivy@grass.poke	과장
5	나무지기	01032540033	sceptile@grass.poke	사원

2. delete 와 select와 random 이용

admin on postgres@hanyangpotal

```

1 delete from grass_corp
2 where phone in(
3     select phone from grass_corp
4     order by random() LIMIT 1);
5 select * from grass_corp

```

	name character varying	phone character varying	email character varying	rank character varying
1	치코리타	01051522001	meganium@grass.p...	사장
2	리피아	01061344185	leafeon@grass.poke	부장
3	주리비안	01064950101	snivy@grass.poke	과장
4	나무지기	01032540033	sceptile@grass.poke	사원

: 임의의 연락처가 한 개 삭제된 것을 확인할 수 있음 table 명만 바꿔 다른 곳에도 적용 가능

- 메일 보내기 기능

1. 한양인 주소록 출력

admin on postgres@hanyangpotal

```
1 select *
2 from 한양인주소록;
3
4 --select *
5 --from grass_corp
```

Data Output

Explain

Messages

Notifications

Query History

	sid character varying	phone character varying	email character varying
10	2012030303	01060294130	leota@hotmail.com
11	2011004533	01080969723	lpaprocki@hotmail.c...
12	2014002331	01043813720	mattie@aol.com
13	2014005004	01096385105	meaghan@hotmail.c...
14	2013004004	01092039494	minna_amigon@yah...
15	2012004203	01094753033	mitsue_tollner@yah...
16	2012394929	01020374564	sage_wieser@cox.net
17	2012004003	01035039847	simona@morasca.co...
18	2014040404	01085931087	wiki_whobrev@aol.c...

2. 그후 개인주소록 출력

1

--select *

2

--from 한양인주소록;

3

4

select *

5

from grass_corp

Data Output

Explain

Messages

Notifications

Query History

	name character varying	phone character varying	email character varying	rank character varying
1	이상해풀	01023140011	ivysaur@grass.poke	이사
2	치코리타	01051522001	meganium@grass.p...	사장
3	주리비안	01064950101	snivy@grass.poke	과장
4	나무지기	01032540033	sceptile@grass.poke	사원
5	리피아	01061344185	leafeon@grass.poke	부장

이때 자신의 개인주소록만 볼 수 있기 위해 from 문에서는 자신이 입력한 개인주소록 파일명의 대한 테이블만 수행한다.

<step 3>

1. domain_name과 local_part로 나누기

admin on postgres@hanyangpotal

```
1 select split_part(h.email,'@',1) as local_part, split_part(h.email,'@',2) as domain_name
2 from 한양인주소록 h
3 --select * from 한양인주소록
```

Data Output Explain Messages Notifications Query History

	local_part text	domain_name text
1	amaclead	gmail.com
2	art	venere.org
3	donette.foll...	cox.net
4	gladys.rim	rim.org
5	kwon	hanyang.ac.kr
6	jbutt	gmail.com
7	josephine_...	darakjy.org
8	kiley.caldar...	aol.com
9	kris	gmail.com
10	leota	hotmail.com
11	lpaprocki	hotmail.com
12	mattie	aol.com
13	meaghan	hotmail.com

: split_part를 통해 @로 local_part와 domain_name을 구분한다.

2. 도수분포표 출력하기

admin on postgres@hanyangpotal

```
1 select split_part(h.email,'@',2) as domain_name, count(*)
2 from 한양인주소록 h
3 group by split_part(h.email,'@',2)
4 --select * from 한양인주소록
```

Data Output Explain Messages Notifications Query History

	domain_name text	count bigint
1	rim.org	1
2	hanyang.ac.kr	1
3	gmail.com	3
4	aol.com	3
5	yahoo.com	2
6	darakjy.org	1
7	venere.org	1
8	morasca.com	1
9	cox.net	2
10	hotmail.com	3